The Thesis Committee for Jonas Reinhardt Michel
Certifies that this is the approved version of the following thesis:

The Gander Search Engine for Personalized Networked Spaces

APPROVED BY

SUPERVISING COMMITTEE:

Christine Julien, Supervisor

Vijay Garg

The Gander Search Engine for Personalized Networked Spaces

by

Jonas Reinhardt Michel, B.S.E.E.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2012

Acknowledgments

I wish to thank my advisor, Dr. Christine Julien, for her invaluable guidance and support and Dr. Jamie Payton and Dr. Gruia-Catalin Roman for their continued collaborative feedback. Support for this work has been provided in part by the National Science Foundation under Grant No. 844850. I am grateful for their support.

The Gander Search Engine for Personalized Networked Spaces

Jonas Reinhardt Michel, M.S.E.
The University of Texas at Austin, 2012

Supervisor: Christine Julien

The vision of pervasive computing is one of a personalized space populated with vast amounts of data that can be exploited by humans. Such *Personalized Networked Spaces* (PNetS) and the requisite support for general-purpose expressive spatiotemporal search of the "here" and "now" have eluded realization, due primarily to the complexities of indexing, storing, and retrieving relevant information within a vast collection of highly ephemeral data. This thesis presents the *Gander* search engine, founded on a novel conceptual model of search in PNetS and targeted for environments characterized by large volumes of highly transient data. We overview this model and provide a realization of it via the architecture and implementation of the Gander search engine. Gander connects formal notions of *sampling* a search space to expressive, spatiotemporal-aware protocols that perform distributed query processing *in situ*. This thesis evaluates Gander through a user study that examines the perceived usability and utility of our mobile application, and benchmarks the performance of Gander in large PNetS through network simulation.

Table of Contents

Acknowledgments **iv**

Abstract **v**

List of Tables **vii**

List of Figures **viii**

Chapter 1. Introduction **1**
1.1 Challenges 2
1.2 Contributions 3

Chapter 2. Motivating Scenarios **5**

Chapter 3. Related Work **7**
3.1 Internet-Based Mechanisms 7
3.2 Wireless Sensor Network Mechanisms 9
3.3 MANET and OppNet Mechanisms 10
3.4 PNetS Search Mechanisms 13

Chapter 4. The Gander Conceptual Model **14**
4.1 A Model of Queries in PNetS 14
4.2 A Model of Data in PNetS 16

Chapter 5. Processing Gander Queries **18**
5.1 Defining Samples for Searches 18
5.2 Secondary Contextual Queries 21

Chapter 6. Implementation **23**
6.1 The myGander Implementation 24
6.2 Connecting to Simulation 25

Chapter 7. Evaluation **27**
7.1 Experimental Settings 27
7.2 Live Gander Queries: A Case Study 28
7.3 Large-Scale Network Evaluation 31

Chapter 8. Implications of Search in PNetS **38**
8.1 Privacy and Security 38
8.2 Incentive for Participation 39
8.3 Storage and Handling of Aged Data 39

Chapter 9. Conclusions and Future Work **41**

Bibliography **42**

List of Tables

7.1 Default Protocol Settings 28

List of Figures

5.1 Query processing styles and sampling 19

6.1 *myGander* Screenshots 23
6.2 *myGander* Architecture 24
6.3 Gander Simulation Architecture 25

7.1 Query Latency and Participation 31
7.2 Query Overhead and Bandwidth 33
7.3 Query Coverage and Distribution 34
7.4 Query Correctness, Coverage, and Distribution 36

Chapter 1

Introduction

The envisioned pervasive computing spaces of the near future will be populated with an immense amount of digital information generated at rapid rates by both humans and digital resources embedded in the environment. Such spaces result from the coalescence of two seemingly disparate trends: (*i*) the increasing ubiquity of sensor-enhanced objects (e.g., buildings [46], cars [25], clothing [20], bicycles [16], plants [49]) and (*ii*) the proliferation of mobile devices (e.g., smart phones, tablets, PDAs). *Personalized Networked Spaces* (PNetS) are pervasive computing environments in which people carry or wear mobile devices that may form opportunistic peer-to-peer connections to each other and to surrounding digital resources. In PNetS, a human user's need to efficiently search the digital space around her, *here* and *now*, is paramount. As our physical surroundings become increasingly digitally accessible, there is an imminent demand for mechanisms that help mobile users find the information they need as they move through densely populated and rapidly changing information spaces.

Tremendous technical resources have been devoted to developing advanced information retrieval techniques that help users find information and

resources (e.g., documents, web pages, social feeds) on the Internet. Users in PNetS, however, need information that is immediate and localized. This tight integration of the user with her immediate surroundings introduces fundamentally different search requirements. An Internet search engine may be used to follow news updates and social feeds about a popular parade. However, when at the parade one might wish to find standing space near friends with a good view or gauge the availability of live video clips of a particular parade float. Using the Internet, one may find available train routes and timetables, whereas a person hurrying to board a crowded train may need to search for the closest available second class seat. When planning a trip to an amusement park, one may use the Internet to find directions, hours of operation, etc. On the other hand, visitors at the park may wish to know which rides have the shortest wait right now or where their friends are. These situations demand expressive capabilities for searching about the here and now; such information is most effectively collected on-demand, directly from the environment. In other words, search *of* the here and now must be performed *in* the here and now.

This paper presents *Gander*—a personalized search engine for the here and now. To our knowledge, Gander is the first search engine for spatiotemporal search of the here and now, in the here and now.

1.1 Challenges

Developing the requisite support for performing search directly within data-rich, dynamic, and unpredictable digital environments calls for a new

paradigm of search that explicitly separates search from advanced indexing of data. This radical shift is motivated by three key factors, which give rise to unprecedented challenges. First, searchable data in PNetS is often supplied by human users, who are likely unwilling to publicly share information (i.e., on the Internet), but willing to share with other nearby, potentially unknown, users [28, 43]. Second, data in PNetS is *ephemeral*, representing changing real-world conditions of the environment, opportunistic social interactions among humans, etc.; information changes on the order of seconds. Moreover, data in PNetS is generated at a rapid rate; there is no expectation it will be consumed. Large volumes of such transient data cannot easily be centrally indexed and associated with a relative spatiotemporal context, as is traditionally done in Internet information retrieval [37]. Finally, there are situations in which Internet access is costly, inconvenient, nonexistent, or simply unnecessary; local interactions may be better managed by exploiting local connectivity, devices that comprise PNetS may be far from reliable connectivity, and connectivity to the Internet may be sparse or financially unattainable. These factors necessitate a decentralized and scalable approach to localized search.

1.2 Contributions

We have previously introduced Gander's conceptual model [39]. In this paper, we leverage and extend this model to realize the Gander search engine. The contributions of this thesis are summarized below. First, we present Gander's architecture and *myGander*, a preliminary search interface for PNetS

devices. This architecture realizes the Gander conceptual model, employs a concrete data model based on global virtual data structures [48], and defines expressive notions of *sampling* for PNetS search supported by existing Mobile Ad Hoc Network routing protocols. Second, we benchmark Gander's sampling approaches and the distributed protocols that support them using a large-scale network simulation driven by real data. Our experiments demonstrate the feasibility and practicality of deploying Gander in large PNetS. Third, we conduct a case study of Gander's usability and utility with human users of *myGander* connected to a simulated PNet driven by data from a real-world pervasive computing space. We have previously demonstrated the *myGander* mobile interface in [41]. In this thesis, we describe *myGander*'s implementation in detail and use it to conduct an in-depth user study. This work also appears as a technical report [40].

Chapter 2

Motivating Scenarios

Myriad opportunities exist for searching for digital information in data-rich, dynamic, and unpredictable surroundings. Consider the following illustrative examples:

While planning a trip to an amusement park, one may use the Internet to find directions, hours of operation, etc. Information needs in the park are dramatically different. A visitor may ask which rides have the shortest wait right now, where the nearest uncrowded restroom is, others' opinions of rides or live performances, etc. Search results may be elicited from both sensing devices embedded in the environment and other users' mobile devices. The data generated by these devices exhibits spatial and temporal locality; it is relevant to users nearby, but not to users far away in space and time. The volume of data generated by devices in the park is too great to be shipped and stored centrally, the lifespan of data is short, the ratio of data used to data available is minute, and visitors may not want to expose data to the Internet. For each of these reasons, information about "here and now" can be better provided by dynamically formed networks in the park. The following queries may be satisfied by the PNet model, but are not easily performed in the Internet:

- *Where is the closest available bench in the shade?*
- *Are there any available lockers close to my location?*

Consider a smaller scale social application that is also dynamic but with data that should not be shared on the Internet:

Classroom experiences are increasingly interactive, including cooperative real-time assignments (e.g., live programming assignments). While traditional material may be most easily accessed via standard archives, the classroom knowledge base (i.e., the knowledge resident in the students about particular tasks and potentially successful strategies) lies in the students (and their devices) themselves. Providing this information via the Internet may be feasible, though it may not be desirable—students may not be comfortable making this information "permanently" available, for example—and is not reflective of the dynamic localized task. Requests about the current collaborative task should be answered "here" and "now" by those engaged in the task:

- *As a student, which other student(s) in the room should I team with to accomplish a challenging task?*
- *As the professor, what are good teams to foster (based on progress and proximity)?*

These situations demand expressive capabilities for searching about the *here* and *now*; such information is most effectively collected on-demand, directly from the environment.

Chapter 3

Related Work

The Gander search engine is motivated by the imminent demand for search mechanisms that enable human users to find information in rapidly changing information rich environments. Yet, enabling efficient access to resources embedded in the environment is a, if not *the*, fundamental component of almost every pervasive computing application. Indeed, countless mechanisms have been developed to mitigate and simplify the challenges that arise due to the inherent dynamics of pervasive computing environments. This section classifies the state of the art search, discovery, and data access mechanisms according to (*i*) the degree of network dynamics and (*ii*) the volume and anticipated volatility of information within the following fundamental pervasive computing application spaces: the Internet, Wireless Sensor Networks, and Mobile Ad Hoc and Opportunistic Networks.

3.1 Internet-Based Mechanisms

Internet-based mechanisms can rely on a high quality of service and operate over relatively static data (e.g., web pages) unlike PNetS where a reliable communication infrastructure does not exist and data is ephemeral.

Google Now[1] incorporates a user's *here* and *now* by automatically retrieving context-sensitive information (e.g., traffic conditions, public transit schedules, nearby restaurant suggestions, etc.) on a user's mobile device. Proponents of the Internet of Things (IoT) [6] and Web of Things (WoT) [23] envision this style of resource access to extend to the mass diffusion of smart objects [6] and web-accessible sensors. The GSN [2] and SenseWeb [30] infrastructures, for example, support keyword search for sensors in a large-scale Internet-based network of heterogeneous sensors and sensor networks. Dyser [47] is a WoT search engine that supports real-time search for sensors with a *dynamic* user-specified state (e.g., a public bike rental station with available bikes). Similarly, the W4 model [12] enables acquisition of digital information about a user's surrounding context from geo-tagged smart objects and Internet-extending sensor networks to support "browsing of the world." Like Gander, these approaches are motivated by human information needs in digitally-accessible environments, but they rely on Internet connectivity and centralized storage resources. Gander targets scenarios in which Internet access may be unattainable, unavailable, or simply undesirable. Therefore, Gander's search mechanisms must exploit local device interactions and distributed resources in a user's immediate environment to support information retrieval within a rapidly evolving information space.

[1] `http://www.google.com/landing/now/`

3.2 Wireless Sensor Network Mechanisms

Efficient and effective data acquisition is paramount in wireless sensor networks (WSNs) where power-, storage-, and computationally-constrained devices must operate for long periods without human intervention. Traditionally, WSN nodes are deployed and opportunistically form a routing structure so as to periodically report their raw sensed data to a fixed base station for centralized processing. Model driven data acquisition [54] significantly improves upon this approach by instead communicating and updating a *model* of the sensed data, provided that the sensed phenomenon exhibits some level of periodicity. Alternatively, the distributed database paradigm [19, 61] provides access to the network through a query interface, treating the WSN as if it were a cohesive storage space. These approaches employ in-network processing techniques like semantic routing [35], distributed query optimization [10], and spatial clustering [55] to facilitate efficient data acquisition in resolution of a structured query. Search mechanisms for PNetS must likewise be resource-conscious, but must additionally provide structured access to high volumes of extremely dynamic information amidst heavy network churn without guarantees of observable periodicity.

Applications characterized by large volumes of sensed data with high degrees of volatility have prompted the development of *reactive* programming abstractions for WSNs that enable detection and monitoring of phenomena of interest. Logical neighborhoods [42] provide access to dynamically formed groups of nodes satisfying a set of logical constraints (e.g., node characteristics

and communication costs) as a single virtual node. Similarly, Regiment [44] is a functional macroprogramming system for WSNs that enables programmers to specify *region streams*, or representations of spatially-distributed, time-varying collections of node state, at compile time to access collective data from groups of nodes sharing geographic, topological, or logical relationships. These and similar strategies (e.g., [1, 11, 60]) handle high volumes of information by enabling access to nodes or information matching some spatiotemporal *signature* or set of logical constraints. Users in PNetS require on-demand access to arbitrary information without *a priori* specifications of interest.

Motivated by the increasing ubiquity of sensors attached to physical objects in our environments, several recent research efforts have developed search engines that enable a user to find sensor-enhanced objects matching an ad hoc query. Snoogle [59], Microsearch [57], and MAX [29], for example, each help a user search for relevant physical objects that are in her vicinity and carry a static textual description. These systems certainly provide search functionality in the direction of PNetS search mechanisms. However, none support the explicit formulation of complex constraints on static sensor content (let alone dynamic content) required by PNetS users or can mitigate the extreme network dynamics exhibited by PNetS.

3.3 MANET and OppNet Mechanisms

Mobile Ad Hoc Networks (MANETs) and Opportunistic Networks (OppNets) are characterized by unpredictable network dynamics, frequent discon-

nects, and high degrees of node mobility. Contrary to PNetS where data generated vastly outweighs data consumed, some MANET applications can harness users' vested interests in specific content or events. The authors of [24], for example, introduce a technique that increases "popular" data's availability by utilizing nodes' gradually acquired local knowledge about global query distribution to push data corresponding to queries with higher frequencies. The publish/subscribe model has received attention within pervasive computing applications that require efficient event distribution in tandem with high degrees of decoupling, flexibility, and scalability. Existing publish/subscribe-based frameworks parameterize event distribution by context [14, 18], social metrics [13], physical proximity [17], contextual relations [9], temporal properties [56], and degree of interest matching [45], for example. These strategies provide mechanisms that enable event-generated content to propagate towards "interested" hosts. Similarly, rule-based approaches like TOTA [36] fully relieve applications from decisions regarding the physical transport of information and instead provide autonomous data propagation governed by application-specific patterns. This style of data-enabled information dissemination fits applications where there is an expectation that data generated will, in general, be consumed. In PNetS, no such assumption can be made. To the contrary, the ratio of data consumed to data generated in PNetS is minuscule.

As in WSNs, programmers may desire on-demand access to the network as a database through a structured query interface. In distributed database frameworks for MANETs (e.g., [4, 5]), a query is sent to a nearby privileged

node (a directory node) possessing copies of its neighbors' schemas, which then generates an execution strategy and coordinates the collaborative resolution of that query in the network. These frameworks require regular synchronization to maintain consistent schema states and query participation roles across the network. Distributed hash table strategies for MANETs (e.g., Ekta [50], MAD-Pastry [62], and Lanes [32]) instead employ a hash function to map data to particular devices while ensuring that information is uniformly and randomly distributed throughout the network. Hash-based techniques require accurate routing tables and uniform data distribution to perform effectively. Still other approaches attempt to maintain application-layer network overlays (e.g., [64]) to mitigate network dynamics and facilitate low-delay queries. Overlay-based strategies enforce rigid communication protocols that must be meticulously maintained. The maintenance requirements present in schema-, hash-, and overlay-based approaches become a hindrance under simultaneous conditions of heavy node churn, high density of nodes, and large volumes of transient data: precisely those conditions that characterize PNetS.

Finally, agent-based approaches employ mobile software agents that act on behalf of a human user to locate and manage distributed and transiently available contextual data. Software agents may be tasked with discovering available services [53], for example. This style of network access fits the characteristics of PNetS well; Gander queries must act on behalf of a user to retrieve relevant information in an efficient manner, reacting to network conditions, potentially over an extended period of time. Future work may focus

on the potential impact of Gander's strategies on agent-based approaches and the synergy between PNetS and multiagent systems. In this work, we investigate the effects of various query distribution strategies, their effects on result quality, and the impact of the resulting network activity.

3.4 PNetS Search Mechanisms

PNetS represent a fusion of pervasive computing's most challenging network conditions and data characteristics. Extreme network dynamics and large volumes of volatile information, coupled with a user's need for on-demand access to information in her here and now in PNetS, necessitate a new paradigm of search mechanisms specifically tailored for these novel search space requirements. The Gander search engine is a first step in this new paradigm of search mechanisms for pervasive computing.

Chapter 4

The Gander Conceptual Model

Gander performs queries about the here and now in the here and now, using locally available capabilities without the support of an Internet infrastructure. We overview the Gander conceptual model [39] and introduce a structural data model for PNetS, which defines the substrate over which queries execute. The model is essential to a rigorous understanding of the quality with which protocols resolve users' searches, and we use it to develop the Gander search engine, which relies on *sampling* as a fundamental component of search.

4.1 A Model of Queries in PNetS

In PNetS, nodes issue queries that are evaluated using information provided by other nodes. A *data item* provides information about the here and now (e.g., a measure of some condition of the environment) and is associated with meta-data that describes its *situation* (e.g., the device(s) that generated it, the location, a timestamp, or even the data's spatiotemporal dynamics, such as volatility or freshness).

Every valid result must "match" the search, which we refer to as *query*

resolution. A query can also include one or more *constraints*. For example, query resolution may identify data items indicating a *bench*; constraints ensure that benches discovered are in the shade. A *relevance metric* compares valid results to each other. A search for a shady bench could favor closer benches or benches close to ice cream vendors. A query can use multiple relevance metrics evaluated independently or using weighted statistics. A *query processing protocol* distributes a query to the PNet. Gander can use the query's contents to direct query processing; ultimately the goal is to collect and present only data that is most relevant.

Our conceptual model must enable expressive query processing protocols that incorporate constraints and relevance. We rely on *partial functions*, which are not required to be defined for every element of their domains. Partial functions naturally lend themselves to *incremental* query processing, which can consider additional data items as they are discovered. The practical realization of such query processing is described in Section 5. Here, we overview the partial functions that define the conceptual model.

Gander Queries. A query is a partial function $G_h : D \rightarrow \Phi$; D contains all data items, Φ is the domain of relevance, and h is the node issuing the query. In a *data item*, (ν, d), ν is the data value and d is the value's meta-data. Query processing is a partial function $\mathrm{QP_h} : \mathrm{D} \rightarrow \mathrm{D}$. Informally, $\mathrm{QP_h}((\nu, \mathrm{d})) = (\nu, \mathrm{d})$ if $(\nu, d) \in D$ is a "valid" result; otherwise $\mathrm{QP_h}((\nu, \mathrm{d}))$ is undefined. $\mathrm{QP_h}$ is a *filter* on D with three pieces:

Reachability. The partial function $\mathcal{R}_h : D \rightarrow D$ expresses whether (ν, d) is *reachable* from h; if not, $\mathcal{R}_h((\nu, d))$ is not defined. We focus on *query* reachability, the ability to send a query to some h' *and* receive a response [52]; $\mathcal{R}$ depends on actual communication capabilities and the protocols used.

Query Resolution. The partial function $\mathcal{S} : D \rightarrow D$ is defined for each $(\nu, d) \in D$ that matches the search.

Query Constraint. The partial function $\mathcal{C} : D \rightarrow D$ is defined for each $(\nu, d) \in D$ that satisfies the query constraints; $\mathcal{C}$'s resolution may rely on the data item's *meta-data* (d).

These functions filter D to the subset of reachable items that satisfy the search string and constraints.

Relevance. A *relevance metric*, $\mathcal{M}_i : D \rightarrow \phi_i$ gives the distance of a data item (ν, d) from an ideal. A Gander query may entail more than one relevance metric; a Gander query, $G_h = \mathcal{K}_{\{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n\}} \circ \mathcal{C} \circ \mathcal{S} \circ \mathcal{R}_h$, is therefore a partial function that maps valid results onto the multidimensional space $\Phi = \phi_1 \times \phi_2 \times \cdots \times \phi_n$. $G_h((\nu, d))$ is not defined if (ν, d) is not reachable or does not satisfy the search string or constraints; otherwise $G_h((\nu, d))$ is an n-tuple, where field i has the value $\mathcal{M}_i((\nu, d))$.

4.2 A Model of Data in PNetS

Practically, D is not constructed centrally; instead data items in D are generated by and stored at devices distributed in the PNet. From a query's

perspective, D is a *global virtual data structure* [48]; resolving a concrete Gander query requires accessing components of this global structure that are distributed in a dynamic and unpredictable network.

To provide this global virtual data structure, we use a *tuple space* that contains semi-structured data [3, 21]. A data item's ν is a tuple that consists of an unordered set of name/value pairs. Meta-data, d, is treated similarly, but the tuple is constructed on the fly by assessing the instantaneous context. Queries and constraints are represented as *patterns* that restrict a matching tuple's fields and values. Relevance metrics also reference the fields of the tuples but may include multiple patterns and evaluation functions.

Data can be generated, destroyed, changed, and moved arbitrarily; Gander's query model is independent of these processes. We assume that data is generated close in space and time to the phenomenon that it describes. Gander's query mechanics can apply similarly to other data models for PNetS.

Chapter 5

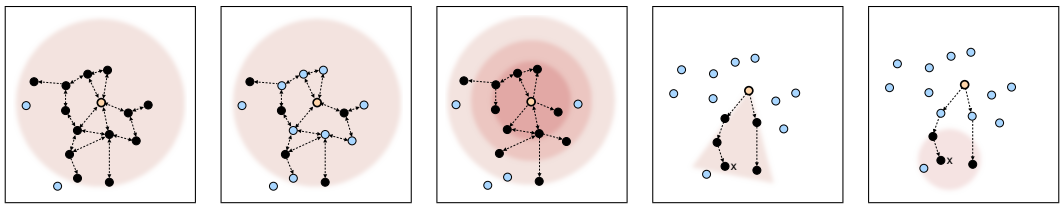
Processing Gander Queries

Acquiring a global view is infeasible in PNetS; protocols must operate only over locally available data. We relate query processing to formal definitions of *sampling*, which enables reasoning about results' quality. We resolve constraints and relevance metrics by inspecting a result's *situation*.

5.1 Defining Samples for Searches

Our partial functions lend themselves to *incremental* protocols, which gradually fill in the functions of G_h. These protocols *sample* a PNet to build a query result Q that represents the desired result G_h. Gander query protocols must efficiently distribute the query (and collect its results) using opportunistic peer-to-peer connections to find the most relevant data.

A variety of strategies exist for opportunistic communication in PNetS. We do not contribute novel protocols; instead, we examine the relationship between existing protocols and the quality of their support for spatiotemporal sampling to identify requirements for the development of novel Gander query protocols. Gander query protocols must provide temporally-sensitive sampling, achieved by processing queries *on-demand*, and spatially-sensitive



(a) (b) (c) (d) (e)

Figure 5.1: Query processing styles and sampling
Dashed lines are sent messages. Darkened nodes respond to a given query. (a) Flooding. Every node in a given range (3 hops) retransmits the query; the target area is the shaded region. (b) Random. A receiving node responds to the query with a given probability; a high quality search evenly samples the shaded space. (c) Probabilistic. Every node that receives the query retransmits it with a given probability; the likelihood of reception drops with distance from the query issuer. (d) Directional. The nodes in the direction of the target location "x" are sampled; a high quality search follows this search "path." (e) Regional. The query targets a location centered at "x;" a high quality search has good coverage of the target region.

sampling, determined by the protocol that selects the space to sample. We quantify spatial quality through *coverage*, which measures how much of the target space the query sampled, and *distribution*, which measures how evenly the query sampled the space. Gander provides five styles of sampling, which tradeoff quality for cost, measured in terms of both latency of search processing and network overhead. Fig. 5.1 shows the styles and their relationships to spatial sampling.

Flooding. These protocols attempt to reach every node; because of the scale of PNetS, we focus on *constrained flooding* in which propagation is limited by network hops (Fig. 5.1(a)). Flooding attempts high *coverage* and reflects the actual evenness (or unevenness) of the nodes' *distribution*.

Random. Random sampling protocols propagate queries similarly to flooding but reduce responses (and the amount of secondary query processing, described below). In random sampling (Fig. 5.1(b)), the likelihood of responding is parameterizable; the goal is to maintain an even *distribution* but reduce *coverage*. Sophisticated sensor network protocols have achieved high-quality uniform spatial samples [8]; our simple protocol can easily be replaced with such a protocol.

Probabilistic. In probabilistic sampling (Fig. 5.1(c)), each node that receives a message probabilistically forwards it; this reduces the overhead, but nodes closer to the issuer are more likely to receive queries [51]. These protocols trade cost for coverage at the edges of the target area, while maintaining even distribution near the query issuer. This style is similar to approaches used for sampling in GIS systems [15].

Directional. Gander's directional sampling (Fig. 5.1(d)) attempts to sample a set of nodes that lie in a direction, given a heading and an angle. The quality of a regional protocol can be judged by how well a forwarding algorithm was able to stay within the designated "path;" again, the goal is good coverage and distribution across the target delivery zone.

Regional. Regional protocols use location to direct packets through the network (Fig 5.1(e)) and aim for good coverage of a specific region. We use location-based routing to reach the target area; a high quality query will provide good coverage and distribution in this space. We effectively com-

bine flooding and a remote space; future work will investigate the use of probabilistic or random sampling with a regional approach.

In Section 7 we evaluate the costs associated with these sampling protocols and the quality of their results; we also relate these quality metrics to metrics that can be measured *on-the-fly* and presented to the query issuer alongside her results.

5.2 Secondary Contextual Queries

Evaluating constraints and relevance entails considering a result's situation, or *context*. A data item's context is captured by its meta-data (d), which can include practically anything: information about the device that "owns" the data, the device's context, other connected devices, their context, etc. Conceptually, d is a snapshot of the PNet from the result's perspective. To compute this context, we need to examine the PNet from the location of the result. Simple context may be available at the device owning the data (e.g., the location or owner's identity). Acquiring more complex context requires a broader network perspective. Rather than always storing (and constantly updating) a personalized view of the PNet for each potential result, we acquire context *on-demand* through secondary *contextual queries* issued from the location of a prospective result. A potential result's owning device can issue a secondary contextual query to evaluate the query's constraints and relevance metrics. Theoretically, secondary queries can use any query processing style,

and they may entail tertiary queries and beyond. In practice, we limit secondary queries to local flooding (within one or two hops) to collect the local situation of a result without burdening the network and delaying the query result.

Chapter 6

Implementation

We next describe the implementation of the *myGander* [41] application, which embodies Gander's search capabilities and its connection with network simulation for large scale evaluation.

(a) (b) (c) (d) (e)

Figure 6.1: *myGander* Screenshots
(a) Search interface, where a user enters search parameters. (b) *myGander* currently allows the user to select from some built-in relevance metrics. (c) The user may select and tailor one of the five query protocols to specify how the PNet should be sampled. (d) *myGander* stores recently issued queries to simplify their recall. (e) Upon completion of a search, *myGander* shows a list of query results that were collected from the live PNet, ranked according to the selected relevance metrics.

6.1 The myGander Implementation

We have implemented *myGander* as a mobile application for the iPhone Operating System (iOS). The decision to initially target iOS was based on the ability to easily create Wi-Fi ad hoc networks with iOS devices without any configuration of the device or operating system (e.g., obtaining root privileges). Fig. 6.1 shows screen shots of *myGander*, which provide the *view* of the model-view-controller pattern of the architecture, shown in Fig. 6.2. The *model* consists of the data and query constructs in Section 4; *controllers* provide an interface of the model components to the view. The model includes a tuple space based data model, a routing table that monitors the PNet's connectivity to maintain updated network routes, and a message center that processes incoming messages (e.g., data items, queries, results).

The model's logic is in the query processor, which resolves received queries using the tuple space, executes secondary queries on the data and its meta-data (collected then stored in the tuple space), and uses the routing table to determine how to route queries and results. *myGander* employs Bonjour[1] to advertise itself, discover locally connected *myGander* de-

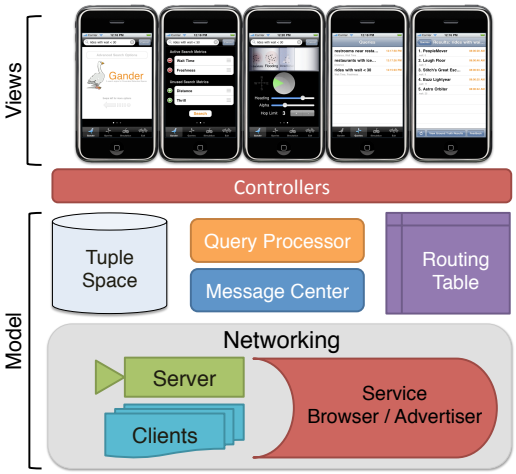


Figure 6.2: *myGander* Architecture

[1]http://www.apple.com/support/bonjour/

vices, and listen for network events using ad hoc network capabilities.

6.2 Connecting to Simulation

myGander provides users with the Gander search engine in real PNetS. Experiments with large-scale PNetS are difficult; thus, we integrated *myGander* and the Gander search engine with the OMNeT++ network simulator [58] (Fig. 6.3). We used OMNeT++'s INET framework [26] for networking and SUMO [33] for node mobility, both of which are publicly available open-source tools. Each simulated node executes Gander's query processing logic, which interacts directly with modified versions of INET's MANET routing capabilities [27] to implement our sampling approaches. We provide baseline implementations of the five protocol styles discussed in Section 5 (i.e., flooding, random, probabilistic, directional, and regional). We use unicast routing to send responses and focus on these generic versions of these standard protocols to ascertain their potential for supporting Gander queries and their quantitative and qualitative differences in succeeding in that support. Indeed, there are many ways to optimize these protocols; future iterations of

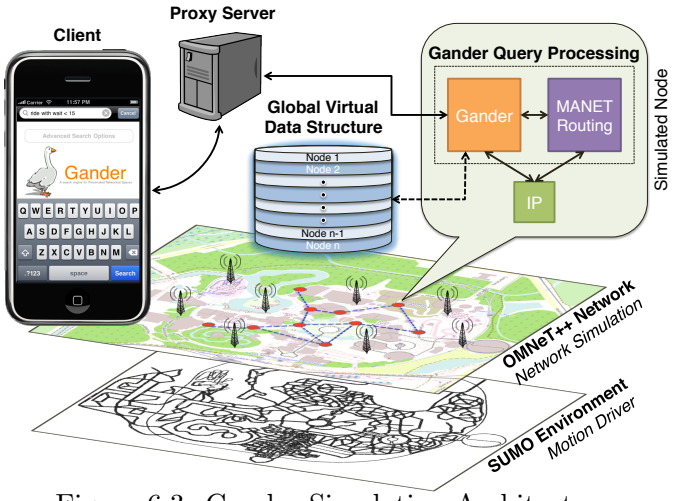


Figure 6.3: Gander Simulation Architecture

Gander will further optimize implementations of each.

During a simulation, each node holds its own piece of the PNet's global virtual data structure (Fig. 6.3); a node uses only this data to process queries. The simulator also allows us to inspect the ground truth and assess the quality of a processed Gander query. *myGander* may be connected to a simulated mobile node through a proxy, which can be set up through the *myGander* interface. This allows live users to interact with large-scale simulated PNetS.

Chapter 7

Evaluation

We evaluate Gander in terms of its usefulness and usability through a study with a simulated large-scale PNet, enabling users of Gander to search a realistic pervasive computing space. We use simulation to benchmark the relative merits of the different Gander query processing styles; our goal is to understand their overarching differences in terms of query latency, overhead, and quality. These results will influence future work on providing optimized and tailored query protocols.

7.1 Experimental Settings

We use *myGander* with an amusement park scenario and real data collected about Disney World's Magic Kingdom, including dynamic wait time information[1] and locations of attractions and amenities. Our PNet includes 30 attractions, 12 restaurants, and 8 restrooms. We populated the park with 1000 users of the *myGander* app (i.e., visitors who store data and participate in query resolution)[2]; users move along park paths following randomly generated

[1]We use ride wait times published by Lines [34]; we collected wait time data every 60 seconds over full day of the park's operation (i.e., 16 hours).

[2]This is roughly 2% of the park's visitors, based on our collected data.

routes at an average speed of 0.5 m/s. Simulated users' devices collect and carry timestamped data about attractions and amenities that they have recently been near (i.e., within ∼20m). In a real PNet, this could correspond to a sensing device embedded in the environment pushing data to a user's device or the user making an observation and creating a piece of human-generated data. Collected data has a 15 minute lifetime, after which it is deleted from the device's tuple space.

The simulated users' devices issue queries, which are distributed using the protocol implementations in Section 6. Table 7.1 lists our default protocol parameters; in the case of both regional and directional, the target was in the center of the park. Secondary queries operate over meta-data, which is also temporarily stored in the local tuple space of the device receiving the query. We determine relevance at the query issuer using the data item and collected meta-data.

Style	**Parameter Settings**
Flooding	hop constraint = 9 hops
Random	response probability = 0.5
	hop constraint = 9 hops
Probabilistic	forward probability = 0.5
	hop constraint = 9 hops
Regional	target distance = 100 m
	region radius = 25 m
Directional	angle = 45 degrees
	hop constraint = 9 hops

Table 7.1: Default Protocol Settings

7.2 Live Gander Queries: A Case Study

We asked users of *myGander*, running on an iPod Touch, to engage with the simulated PNet. We used an in-app *myGander* view to show the participant the ground truth and collect feedback on Gander's performance; all queries in the user study were issued with the flooding protocol.

We tested each participant individually. We asked them to imagine they were visitors to the amusement park and had the ability to ask questions about attractions, their wait times, and other amenities (e.g., eating establishments, restrooms, etc.). Participants were given a complete list of the available data and walked through a sample search to demonstrate Gander's fundamental capabilities: a search for rides, with a constraint of wait times less than 15 minutes, and the combination of a freshness and a distance relevance metric. We then asked participants to formulate their own queries, thinking aloud about the process, their actions, and their impressions of the results. In addition to delivering the query results via the search engine interface, we used the in-app view to show participants the ground truth and asked them to reflect on the quality of the Gander search query.

We gave participants a pre-test to assess demographics and familiarity with related technologies and a post-test to evaluate impressions of Gander and its potential. We studied 12 participants: ten men and two women, of ages ranging from 24 to 51. Participants were solicited from the general public and from the university; they all rated themselves as average or above average with respect to technological savviness in comparison to their peers. All but two participants said they use Internet tools to search for local information, and all but the same two and one other said they regularly used a smart phone or handheld device. The participants were familiar with existing location-based search tools, including specialized applications like those discussed in Section 3.

When asked to rate their agreement with the statement *I would be interested in using a tool like Gander in the future* on a four point scale, eight participants strongly agreed, while the other four agreed. When asked to rate the statement *I found the Gander search interface easy to use* on the same scale, four participants strongly agreed, while the other eight agreed.

When asked if Gander performed better than expected on any of their queries, three participants answered affirmatively. The following statement indicates that the participants were surprised at the level of detail available: "*Rides info can show exactly how many people are waiting in line.*" On the other hand, when asked if Gander performed worse than expected on any queries, six participants answered affirmatively. These discrepancies were commonly due to the quality of the result, e.g., "*The ride search where I set a* > 7 *constraint on thrill factor returned one result where the ground truth results showed nine and included lots that were not in the Gander results.*"[3] or the influence of relevance metrics on displayed results, e.g., "*Ranking could be more accurate.*"[4] The former motivates the need for Gander queries to self-assess their quality and report quality metrics alongside the search results, while the latter demands a careful examination of personalized notions of relevance. We examine both of these questions in more detail in the second piece of our evaluation.

[3]The incomplete Gander results that elicited this comment were due to a sparseness of simulated nodes at this participant's query's location.

[4]This comment was made upon the participant realizing that the app didn't support re-ranking acquired results without issuing an entirely new query with new relevance metrics.

7.3 Large-Scale Network Evaluation

For our large scale network evaluation, we use six similar but increasingly complex queries. The queries have the same search string ("thrill ride") and constraint ("with wait time less than 20 minutes") but different relevance metrics: **Q1:** wait time; **Q2:** distance from me; **Q3:** thrill; **Q4:** wait time then distance from me; **Q5**: wait time then thrill; **Q6:** thrill then wait time. Queries with multiple relevance metrics use lexicographical ordering; evaluating the expressiveness of more complex combinations is beyond the scope of this paper. We average results across these six queries. In our simulations, each query issuer issues one query every minute; we present averages over all of the queries issued in four hours. Unless otherwise specified, 20% of the nodes (i.e., 200 users) are designated query issuers.

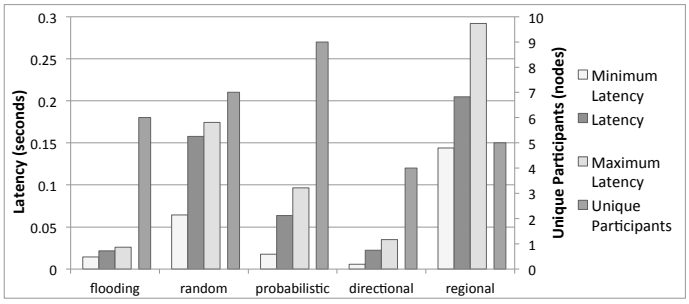


Figure 7.1: Query Latency and Participation

We relied on existing communication stack behavior below the network layer and built our query dissemination styles on top of the existing MANET routing protocols in the simulator. We used INET's implementation of 802.11. We randomly select nodes in the simulation to issue queries, and we vary the number of simultaneous query issues to evaluate Gander's scalability with respect to increased query traffic.

We first benchmark the styles in terms of query latency and participation (Fig. 7.1). We report the average minimum latency (the time to receive the first result), the average latency of all received results, the average maximum latency (the time to receive the last query result), and the average unique number of nodes that participate in a query[5]. Not surprisingly, the regional style has the largest latencies of the five sampling styles since queries issued with this style must first reach the target area (the center of the park in this case) before eliciting results. Perhaps unexpectedly, flooding has the lowest latencies of the three styles that sample a circular region around the issuer. Intuitively, one would expect this style to produce the largest average and maximum latencies since it stipulates the sampling of all devices in the sample space. These abnormally low latencies can be attributed to link contention. For a flooding query, all nodes receiving a query immediately send replies, which interferes with the continuing query propagation and causes congestion, necessitating retransmissions. These low flooding latencies reflect a query's inability to propagate more than a few hops in the sample space. This conclusion is corroborated by the relatively low number of query participants for flooding, which one would expect to be higher than both random and probabilistic.

Next, we benchmark the sampling styles in terms of query overhead and bandwidth (Fig. 7.2). We report the average per-query overhead of distributing a query, successfully and unsuccessfully, and of sending the responses measured

[5]Unless otherwise stated, we use medians for averages, which helps in identifying trends when there are a few significant outliers

in number of messages and bytes. The data labels in Fig. 7.2 indicate the ratio of result overhead to total query overhead, which can be thought of as a measure of a style's "payoff." Flooding requires dramatically lower overhead than random and probabilistic (one would expect the opposite), supporting the conclusion that congestion has limited the propagation of the flooded messages.

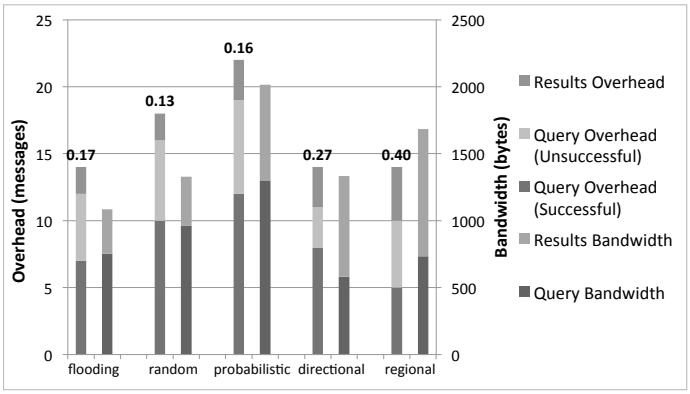


Figure 7.2: Query Overhead and Bandwidth

Flooding, random, and probabilistic exhibit relatively identical payoffs. Both the directional and regional styles seem to exhibit a higher degree of payoff, requiring fewer queries to obtain a comparable number of result messages, but one must take care when interpreting these results. These two sampling styles target a particular direction or location; however, the query is not necessarily representative of this direction or location of interest. Therefore, directional and regional styles essentially reduce the sample space and the number of queries sent, producing an increase in a query's perceived payoff.

As made apparent in our user study, the quality of Gander queries is key. We compute *coverage* of both the data returned and the nodes that responded. We compute the target area for flooding, random, and probabilistic as a circle centered at the query issuer with a radius equal to the protocol's

hop constraint multiplied by 20m, our effective wireless range. For directional, we compute the target area as an isosceles triangle extending from the query issuer towards the middle of the park with the equal sides separated by an angle of 45 degrees and an altitude of length equal to the protocol's hop constraint multiplied by 20m. The target area for regional is circle with a radius of 25m centered at the a point 100m from the query issuer towards the middle of the park. We associate each data item (or responder) with a circular area of radius 20m and define coverage as the percentage of overlap between the areas associated with the data items returned as a result (or the areas associated with the responding nodes) and the target area.

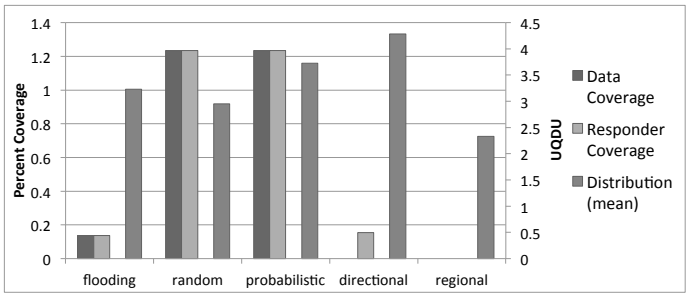


Figure 7.3: Query Coverage and Distribution

We compute *distribution* using an upper quartile distribution uniformity (UQDU) test [31]. This test divides the target area into equal-sized bins, counts the nodes in each bin, and divides the density of the 75th percentile of bins by the expected density for the target area. A uniform distribution results in a value of 1; the further from 1 the UQDU varies, the less uniform a distribution is. Fig. 7.3 plots the coverage and distribution (i.e., distance from 1) achieved by the five sampling styles; a lower UQDU indicates a more uniform distribution.

The supposed congestion induced by the flooding protocol is further

corroborated by the poor coverage it is shown to achieve here. Random and probabilistic achieve nearly identical coverage, which makes sense as they share identical sampling spaces, but the random style samples in a more evenly distributed manner. Neither directional nor regional achieve good coverage, likely because the majority of their sample space is too disconnected from the query issuer's "here" to return a significant number of results. The directional style's distribution is particularly poor as well. This is probably because a directional query issuer is more likely to successfully receive responses from neighbors that are closer in the sample space, resulting in a tightly packed group of responders (i.e., unevenly distributed with respect to the whole sample space).

We can dynamically assess freshness, coverage, and distribution as part of a Gander query and present them to users as descriptors of search quality. To justify these quality metrics, we relate them to an omniscient perspective on query correctness. We use the Jaccard index, which, given an ideal Gander query, G_h, and results of a processed query, Q, is: $\frac{|Q \cap G_h|}{|Q \cup G_h|}$. Figs. 7.4(a)-(c) plot this correctness versus coverage and distribution for flooding; the other protocols had similar results.

Figs. 7.4(a) and (b) illustrate that, even in the presence of severe network congestion, coverage is positively correlated with correctness, making it meaningful as user feedback regarding quality. This result is intuitive though. As shown in Fig. 7.4(d), when responder coverage is increased more nodes process Q providing results and more data is acquired producing a result set

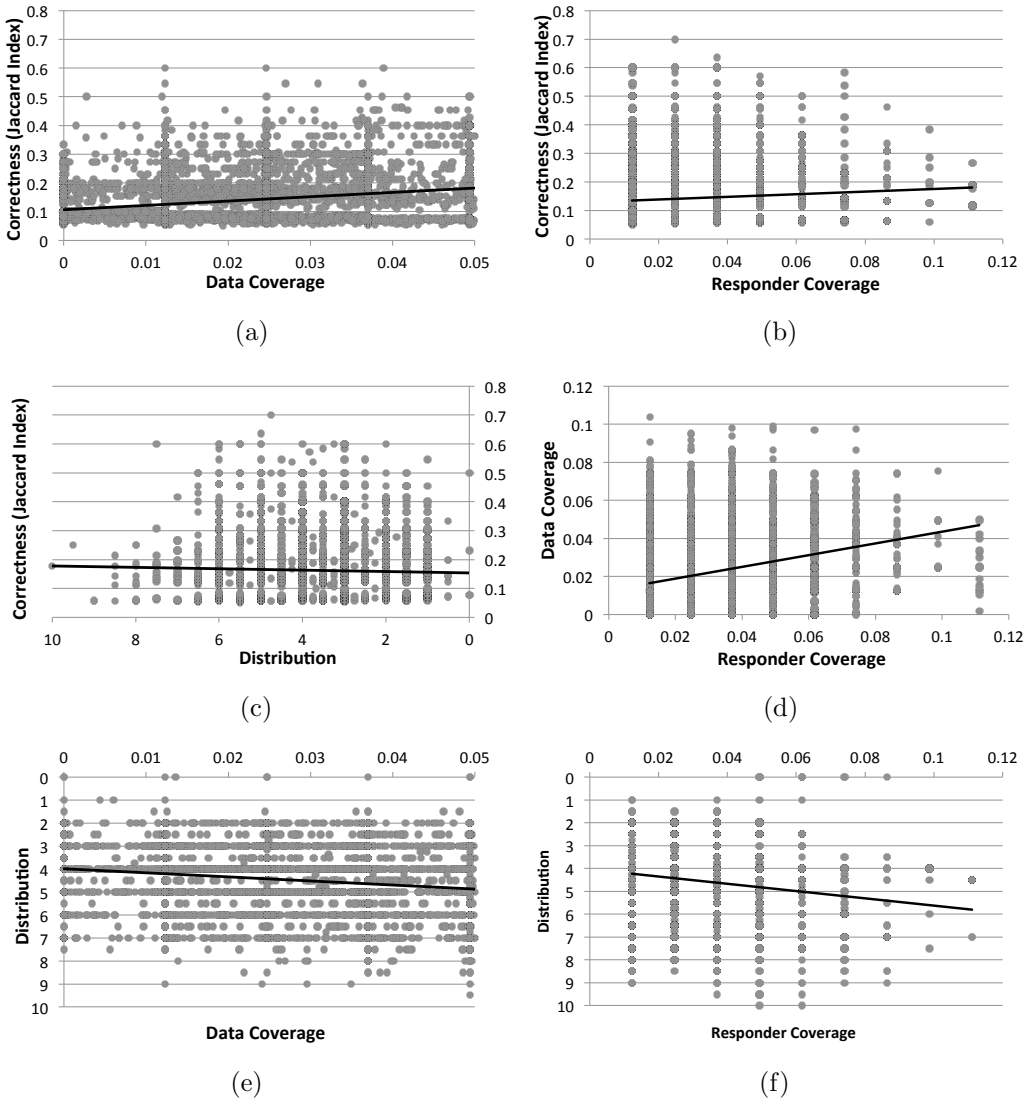


Figure 7.4: Query Correctness, Coverage, and Distribution

closer to G_h (i.e., more correct). We also compared freshness to the Jaccard index; the correlation between freshness and correctness exists but is not as strong; we omit this chart for brevity.

Distribution, on the other hand, appears to be negatively correlated with correctness (Fig. 7.4(c)). This is because a better distribution of Q does

not necessarily provide better coverage. In fact, Figs. 7.4(e) and (f) indicate the opposite; distribution is poorer for higher data and responder coverage. These results motivate Gander's need for reflective and adaptive query processing protocols that consider the state of the PNet in relation to Q, so as to provide high quality results without over-taxing the network.

Chapter 8

Implications of Search in PNetS

The realization of a search engine for PNetS has potential to impact a number of important networking and information concerns. This section identifies and discusses the broader implications of Gander and offers some preliminary suggestions to mitigate several immediately apparent issues.

8.1 Privacy and Security

Perhaps the most prominent concern of search mechanisms for PNetS is that of privacy and security. Data generated and acquired in PNetS is inherently most relevant in the time and place in which it is generated and therefore possesses an innate ability to expose sensitive information about an environment or even users in an environment. Furthermore, the introduction of falsified information could produce adverse effects. The development of privacy and security mechanisms for Gander remains an important component of future work. The authors of [59] describe a distributed security framework that determines access to data objects using rules akin to permissions in a UNIX file system. A similar approach could provide privacy guarantees in a PNetS search engine. To mitigate the effects of falsified information, Gander

could associate a degree of *trust* with data sources in the future (e.g., using a reputation scheme [24] or a trust model [63]).

8.2 Incentive for Participation

One important concern, not necessarily specific to a PNetS search engine, is the issue of participation and incentive; users need valid reasons to lend their devices' energy, bandwidth, CPU, and storage resources to others' PNetS traffic. This in fact is a concern for any MANET or peer to peer (P2P) application; both MANETs and P2P networks require nodes assist one another to make the network truly useful. Likewise, Gander's efficacy and quality of service (QoS) are highly correlated with the level of participation of users with others' queries within PNetS. While an effective incentive model is out of scope of this work, several reputation-, credit-, and exchange-based incentive schemes for MANETs have been proposed that could be useful means of explicitly ensuring sufficient participation. We direct readers to [38] for an excellent review of these MANET incentive schemes. Alternatively, with the realization of large-scale opportunistic and P2P wireless infrastructures (e.g., FlashLinQ [7]) it would be entirely feasible to piggyback Gander traffic on existing network traffic to provide sufficient QoS.

8.3 Storage and Handling of Aged Data

PNetS possess the ability to generate massive amounts of information. As time passes, devices move, and information spreads (e.g., through the trans-

mission of queries and results), data may decay both in space and time, losing its potency of relevance. Devices comprising PNetS are resource-constrained, having limited storage space; therefore, information cannot be stored indefinitely. In this work, devices simply delete information after it reaches a certain age. However, the storage and handling of aged and decayed data in PNetS search mechanisms is an important concern we plan to address in future work. Specifically, we intend to explore how Gander may employ a trajectory-based data model to enable application-, device-, or data-specific policies that govern when and how data should decay and finally be destroyed.

Chapter 9

Conclusions and Future Work

We introduced, implemented, and evaluated the Gander search engine and demonstrated that a fundamental shift in the conceptual model of search is necessary and feasible in practical pervasive computing networks. This paper enables such a shift, contributing the first approach for search *of* the here and now, *in* the here and now. We evaluated the relative merits of query processing styles; optimizing protocols *within* styles will be an important step in Gander's practical implementation. Similarly, secondary contextual queries demand more sophisticated processing. Proactively collecting and processing commonly used contextual data may lower both overhead and latency (as demonstrated in [22]); augmenting this with knowledge about other users and their searches, providing a kind of PNetS collaborative filtering, could aid this process. Finally, a user in our study commented that in an ideal world, "*search items would be related to their own search metrics automatically.*" Such intelligence is a key future objective for the Gander search engine.

Bibliography

[1] T. Abdelzaher, B. Blum, Q. Cao, Y. Chen, D. Evans, J. George, S. George, L. Gu, T. He, S. Krishnamurthy, L. L. Luo, S. Son, J. Stankovic, R. Stoleru, and A. Wood. Envirotrack: towards an environmental computing paradigm for distributed sensor networks. In *ICDCS*, pages 582–589, 2004.

[2] K. Aberer, M. Hauswirth, and A. Salehi. Infrastructure for data processing in large-scale interconnected sensor networks. In *MDM*, pages 198–205, 2007.

[3] S. Abiteboul. Querying semi-structured data. In *ICDT*, 1997.

[4] A.M. Amja, A. Obaid, and N. Seguin. A distributed mobile database architecture. In *APSCC*, pages 62 –69, 2011.

[5] H. Artail, H. Safa, R. ELZinnar, and H. Hamze. A distributed database framework from mobile databases in manets. In *WiMOB*, 2007.

[6] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer Net.*, 54(15):2787–2805, 2010.

[7] F. Baccelli, N. Khude, R. Laroia, J. Li, T. Richardson, S. Shakkottai, S. Tavildar, and X. Wu. On the design of device-to-device autonomous discovery. In *COMSNETS*, pages 1–9, 2012.

[8] B. A. Bash, J. W. Byers, and J. Considine. Approximately uniform random sampling in sensor networks. In *DMSN*, 2004.

[9] R.T. Boyer and W.G. Griswold. Fulcrum - an open-implementation approach to internet-scale context-aware publish / subscribe. *Hawaii International Conference on System Sciences*, 9:275a, 2005.

[10] I. Brunkhorst, H. Dhraief, A. Kemper, W. Nejdl, and C. Wiesner. Distributed queries and query optimization in schema-based p2p-systems. In *Databases, Inf. Sys., and P2P Comp.*, volume 2944 of *Lecture Notes in Computer Science*, pages 184–199. Springer Berlin / Heidelberg, 2004.

[11] Q. Cao and T. Abdelzaher. Scalable logical coordinates framework for routing in wireless sensor networks. *ACM Trans. Sen. Netw.*, 2:557–593, 2006.

[12] G. Castelli, A. Rosi, M. Mamei, and F. Zambonelli. A simple model and infrastructure for context-aware browsing of the world. In *PerCom*, pages 229 –238, 2007.

[13] P. Costa, C. Mascolo, M. Musolesi, and G.P. Picco. Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks. *IEEE J. on Selected Areas in Comm.*, 26(5), 2008.

[14] G. Cugola, A. A. Margara, and M. Migliavacca. Context-aware publish-subscribe: Model, implementation, and evaluation. In *ISCC*, 2009.

[15] X. Dai, M. L. Yiu, N. Mamoulis, Y. Tao, and M. Vaitis. Probabilistic spatial queries on existentially uncertain data. In *SSTD*, 2005.

[16] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell. Bikenet: A mobile sensing system for cyclist experience mapping. *ACM Trans. Sen. Netw.*, 6(1):6:1–6:39, 2010.

[17] L. Fiege, F.C. Gartner, O. Kasten, and A. Zeidler. Supporting mobility in content-based publish/subscribe middleware. In *Middleware*, volume 2672 of *Lecture Notes in Computer Science*, pages 103–122. Springer Berlin / Heidelberg, 2003.

[18] D. Frey and G.C. Roman. Context-aware publish subscribe in mobile ad hoc networks. In *Coordination Models and Languages*, volume 4467 of *Lecture Notes in Computer Science*, pages 37–55. Springer Berlin / Heidelberg, 2007.

[19] I. Galpin, C. Brenninkmeijer, A. Gray, F. Jabeen, A. Fernandes, and N. Paton. Snee: a query processor for wireless sensor networks. *Dist. and Parallel Databases*, 29:31–85, 2011.

[20] R. K. Ganti, P. Jayachandran, T. F. Abdelzaher, and J. A. Stankovic. SATIRE: a software architecture for smart attire. In *MobiSys*, MobiSys '06, pages 110–123, 2006.

[21] D. Gelernter. Generative communication in linda. *ACM Trans. Program. Lang. Syst.*, 7(1), 1985.

[22] E. Grim, C. Fok, and C. Julien. Grapevine: Efficient situational awareness in pervasive computing environments. In *PerCom (WIP Track)*, pages 475 –478, 2012.

[23] D. Guinard and T. Vlad. Towards the web of things: web mashups for embedded devices. In *WWW*, 2009.

[24] S. Gupta, A. Joshi, J. Santiago, and A. Patwardhan. Query distribution estimation and predictive caching in mobile ad hoc networks. In *MobiDE*, pages 24–30, 2008.

[25] B. Hull, V. Bychkovsky, K. Chen, M. Goraczko, A. Miu, E. Shih, Y. Zhang, H. Balakrishnan, and S. Madden. CarTel: A distributed mobile sensor computing system. In *SenSys*, SenSys '06, pages 125–138, 2006.

[26] The INET Framework for OMNeT++. `http://inet.omnetpp.org/`, 2012.

[27] The INETMANET Framework for OMNeT++. `https://github.com/inetmanet/inetmanet/wiki`, 2012.

[28] Q. Jones, S.A. Grandhi, S. Karam, S. Whittaker, C. Zhou, , and L. Terveen. Geographic place and community information preferences. *CSCW*, 17(2–3):137–167, 2008.

[29] K.-K.Yap, V. Srinivasan, and M. Motani. Max: Human-centric search of the physical world. In *SenSys*, pages 166–179, 2005.

[30] A. Kansal, S. Nath, J. Liu, and F. Zhao. Senseweb: An infrastructure for shared sensing. *IEEE MultiMedia*, 14(4):8–13, 2007.

[31] D. Kieffer and T.S. O'Connor. Managing soil moisture on golf greens using a portable wave reflectometer. In *Int. Irrig. Show*, December 2007.

[32] M. Klein, B. König-Ries, and P. Obreiter. Lanes – a lightweight overlay for service discovery in mobile ad hoc networks. Technical Report 2003-6, University of Karlsruhe, 2003.

[33] D. Krajzewicz, G. Hertkorn, C. Rössel, and P. Wagner. SUMO (simulation of urban mobility): An open-source traffic simulation. In *MESM*, 2002.

[34] Disney World Lines App (Touring Plans). `http://www.touringplans.com/walt-disney-world-lines`.

[35] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *ACM SIGMOD*, pages 491–502, 2003.

[36] M. Mamei, F. Zambonelli, and L. Leonardi. Tuples on the air: a middleware for context-aware computing in dynamic networks. In *ICDCS*, pages 342–347, 2003.

[37] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[38] A. Mawji and H. Hassanein. Incentives for p2p file sharing in mobile ad hoc networks. In *CCNC*, pages 1 –5, 2009.

[39] J. Michel, C. Julien, J. Payton, and G.-C. Roman. Gander: Personalizing search of the here and now. In *MobiQuitous*, 2011.

[40] J. Michel, C. Julien, J. Payton, and G.-C. Roman. The gander search engine for personalized networked spaces. Technical Report TR-ARiSE-2012-009, The University of Texas at Austin, November 2012.

[41] J. Michel, C. Julien, J. Payton, and G.-C. Roman. myGander: A mobile interface and distributed search engine for pervasive computing. In *PerCom (Demonstrations Track)*, pages 495 –497, 2012.

[42] L. Mottola and G.P. Picco. Programming wireless sensor networks with logical neighborhoods. In *InterSense*, 2006.

[43] D. Mountain and A. MacFarlane. Geographic information retrieval in a mobile environment: evaluating the needs of mobile individuals. *J. of Info. Science*, 33:515–530, 2007.

[44] R. Newton and M. Welsh. Region streams: functional macroprogramming for sensor networks. In *DMSN*, pages 78–87, 2004.

[45] E. Nordström, P. Gunningberg, , and C. Rohner. A search-based network architecture for mobile devices. Technical Report 2009-003, Uppsala University, January 2009.

[46] C. Ono and A. Mita. Genetic mechanism for designing new generation of buildings from data obtained by sensor agent robots. In *SPIE Smart Structures/NDE 2012*, 2012.

[47] B. Ostermaier, K. Römer, F. Mattern, M. Fahrmair, and W. Kellerer. A real-time search engine for the web of things. In *IOT*, 2010.

[48] G. P. Picco, A. Murphy, and G.-C. Roman. On global virtual data structures. In *Process Coordination and Ubiquitous Computing*. 2002.

[49] I. Poupyrev, O. Schoessler, J. Loh, and M. Sato. Botanicus interacticus: Interactive plant technology. In *SIGGRAPH*, 2012.

[50] H. Pucha, S.M. Das, and Y.C. Hu. Ekta: an efficient dht substrate for distributed applications in mobile ad hoc networks. In *WMCSA*, pages 163 – 173, 2004.

[51] V. Rajamani and C. Julien. Adaptive data quality for persistent queries in sensor networks. In *QShine*, 2009.

[52] V. Rajamani, C. Julien, J. Payton, and G.-C. Roman. Inquiry and introspection for non-deterministic queries in mobile networks. In *FASE*, pages 401–416, 2009.

[53] O. Ratsimor, D. Chakraborty, A. Joshi, and T. Finin. Allia: alliance-based service discovery for ad-hoc environments. In *WMC*, pages 1–9, 2002.

[54] U. Raza, A. Camerra, A. L. Murphy, T. Palpanas, and G. P. Picco. What does model-driven data acquisition really achieve in wireless sensor networks? In *PerCom*, pages 84–94, 2012.

[55] R. Rosemark and W.-C. Lee. Decentralizing query processing in sensor networks. In *MobiQuitous*, pages 270 – 280, 2005.

[56] G. Sollazzo, M. Musolesi, and G. Mascolo. Taco-dtn: a time-aware content-based dissemination system for delay tolerant networks. In *MobiOpp*, pages 83–90, 2007.

[57] C. Tan, B. Sheng, H. Wang, and Q. Li. Microsearch: When search engines meet small devices. In *Pervasive Computing*, volume 5013 of *Lecture Notes in Computer Science*, pages 93–110. Springer Berlin / Heidelberg, 2008.

[58] A. Vargas. OMNeT++ Web Page. `http://www.omnetpp.org`, 2008.

[59] H. Wang, C. C. Tan, and Q. Li. Snoogle: A search engine for pervasive environments. *IEEE Trans. on Parallel and Dist. Sys.*, 21(8):1188–1202, 2010.

[60] K. Whitehouse, C. Sharp, E. Brewer, and D. Culler. Hood: A neighborhood abstraction for sensor networks. In *MobiSys*, pages 99–110, 2004.

[61] Y. Yao and J. Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Rec.*, 31(3):9–18, 2002.

[62] T Zahn and J. Schiller. Madpastry: A dht substrate for practicably sized manets. In *ASWN*, 2005.

[63] L. Zhaoyu, A.W. Joy, and R.A. Thompson. A dynamic trust model for mobile ad hoc networks. In *FTDCS*, pages 80 – 85, 2004.

[64] A. Ziotopoulos and G. de Veciana. P2P network for storage and query of a spatio-temporal flow of events. In *PerCom Workshops*, 2011.